AI-140-16-Modbus

Analog Input to Modbus RTU Gateway

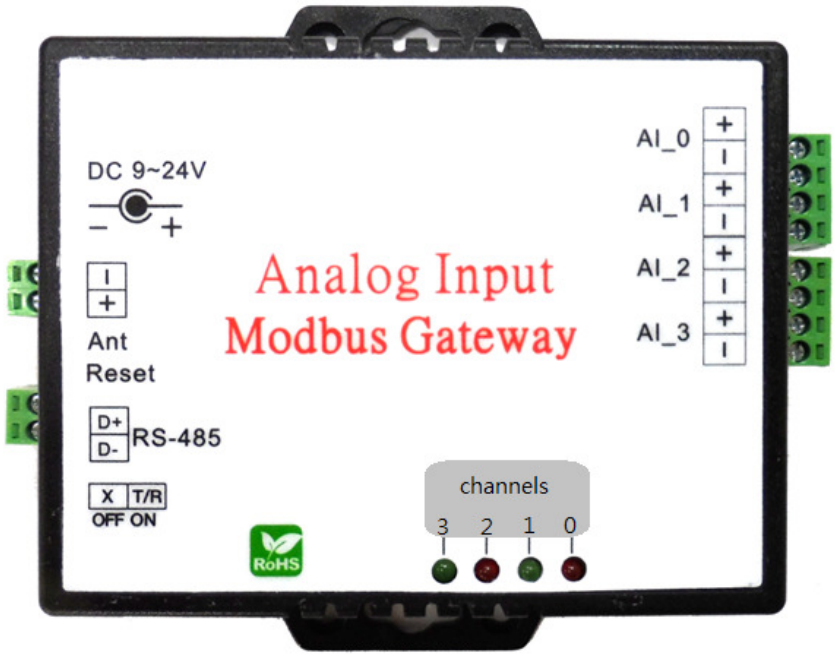


TABLE OF CONTENTS

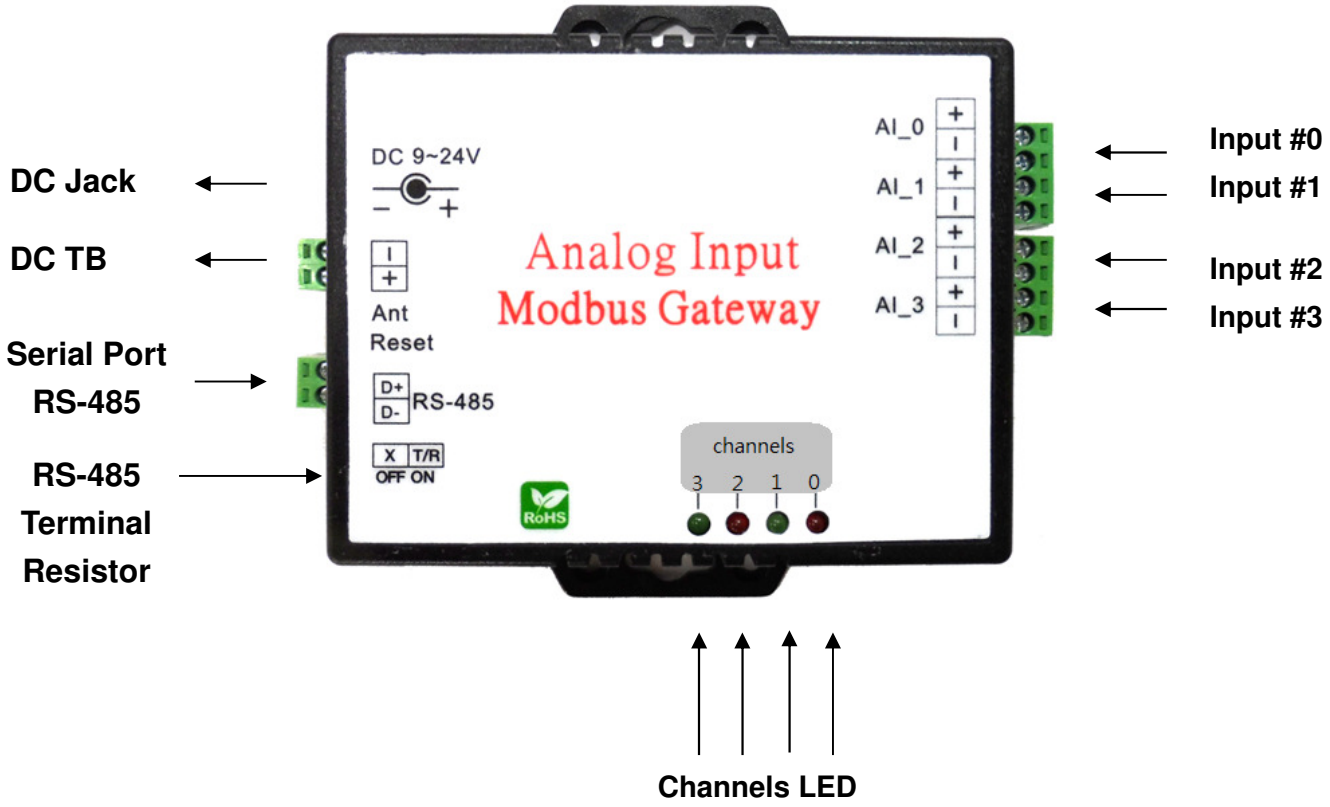
Introduction

This Analog Input to be connected via RS-485 as a remote unit. It provides 4 channels for input with mA or DCV can be preset upon ordering or adjusted the Jumpers on PCB as an option. It will reply digital figures of mA or DCV upon receiving Modbus query from Modbus Master (host computer).

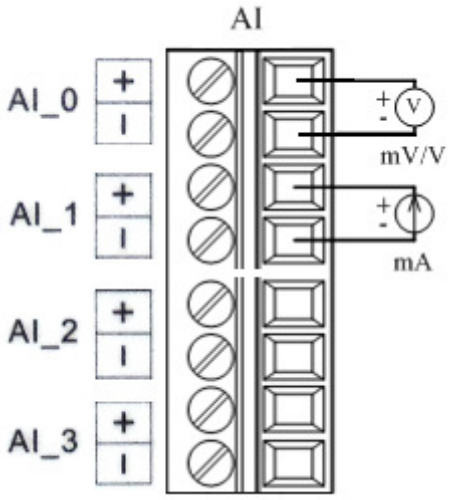
It is easily configured via a Utility Tool which can set ID, Baud Rate, Data Bits, Parity and Stop Bits in order to improve the efficiency. The circular loop test via “autoscan” function provides testing result very quick.

2

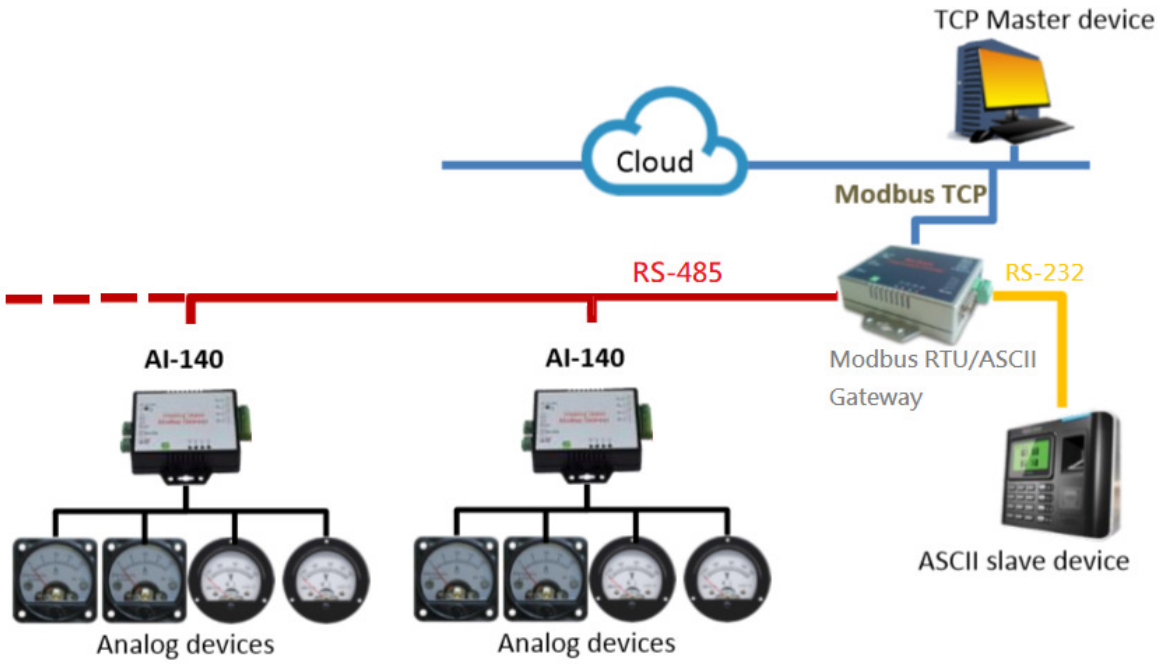
Product Panel Views



2.1 Wiring connection



2.2 Applications



3

Product specifications

- **Models (option)**
  - AI-140-16 ：4～20mA input x2，0～10V input x2
  - AI-140-16A：4～20mA input x4
  - AI-140-16V：0～10V input x4

- **Hardware**
  - CPU：8-bit CPU
  - RAM：64K bytes SRAM
  - ROM：256K bytes Flash ROM

- **Analog input**
  - 16 bits resolution.
  - Support 4～20mA or 0～10DCV (to be set by jumpers)

- **Serial port**
  - RS-485 x1, terminal block
  - Built-in 120-ohm Terminal resistor
  - Baud Rate : 300 bps ~ 115.2 Kbps
  - Parity：none
  - Data Bits：8
  - Stop Bits：1
  - Surge protection

- **Watch Dog**

- **Power：DC9～24 V，200mA@12VDC**
  Connector: DC JACK or Terminal Block
- **LED**：
  AI_0, AI_1, AI_2, AI_3 (4 channels)

- **Environment**
  - Operating temperature：-0℃～+70℃
  - Storage temperature：-10℃～+85℃

- **Dimensions**：110 * 90 * 26 mm（LxWxH）

- **Weight:** 95gm

- **Option: Din Rail parts**

- **Certifications:**
  - CE / FCC
  - RoHS

- **Warranty: 1 year**

4

MODBUS directive and parameters

A basic MODBUS command always takes two main parameters：

- ID Address：Designated to receive this Command ID address。
- Function Code：This Command function。

(Default　ID : 1 , Baud rate : 38400)

The reception to the Command element will return a Response inform the completion of action in response to the remote or the value returned by the read, Response Command format in the format is basically the same, but also have ID Address and Function code in order to master identification, the following table for common Function code order.

MODBUS RTU Protocol

- The following MODBUS functions are supported.

Function code	Description
0x03	Read Holding Registers
0x06	Write Single Register

- MODBUS Protocol Address Map

MODBUS Register	HEX	Function	Description	Action
0000	0106	Read/Write Modbus device Address And Baud Rate Setting	High byte: Modbus device Address The range is 1~247 Low byte: Baud Rate Setting 1: 2400 2: 4800 3: 9600 4: 14400 5: 19200 6: 38400 7: 115200	R/W
0004	0018	Read AI1 voltage	Ex: 0x0018 = 24 in decimal. It means 0.24V	R
0005	0030	Read AI1 current	Ex: 0x0030 = 48 in decimal. It means 0.48mA	R
0008	0018	Read AI2 voltage	Ex: 0x0018 = 24 in decimal. It means 0.24V	R
0009	0030	Read AI2 current	Ex: 0x0030 = 48 in decimal. It means 0.48mA	R
000c	0018	Read AI3 voltage	Ex: 0x0018 = 24 in decimal. It means 0.24V	R
000d	0030	Read AI3 current	Ex: 0x0030 = 48 in decimal. It means 0.48mA	R
0010	0018	Read AI4 voltage	Ex: 0x0018 = 24 in decimal. It means 0.24V	R
0011	0030	Read AI4 current	Ex: 0x0030 = 48 in decimal. It means 0.48mA	R

- Example for read Modbus Address and Baud Rate :

Request		Response	
Field Name	(Hex)	Field Name	(Hex)
MODBUS Address	01	MODBUS Address	01
Function	03	Function	03
Starting Address Hi	00	Byte Count	02
Starting Address Lo	00	High Byte Data	01
Quantity of Outputs Hi	00	Low Byte Data	06
Quantity of Outputs Lo	01	High Byte CRC	39
High Byte CRC	84	Low Byte CRC	D6
Low Byte CRC	0A		

- **Press Reset Key for over 5 seconds will set device back to default**

Ex: High Byte Data 0x01 means Modbus Address = 01. The available range is 1~247.

Low Byte Data 0x06 means Baud Rate = 38400

For 1 means baud rate is 2400

2 means baud rate is 4800

3 means baud rate is 9600

4 means baud rate is 14400

5 means baud rate is 19200

6 means baud rate is 38400

7 means baud rate is 115200

- Example for read AI1 voltage input

Ex: Data 0x0018 = 24 in decimal. It means 0.24V

Request		Response	
Field Name	(Hex)	Field Name	(Hex)
MODBUS Address	01	MODBUS Address	01
Function	03	Function	03
Starting Address Hi	00	Byte Count	02
Starting Address Lo	04	High Byte Data	00
Quantity of Outputs Hi	00	Low Byte Data	18
Quantity of Outputs Lo	01	High Byte CRC	B8
High Byte CRC	C5	Low Byte CRC	4E
Low Byte CRC	CB		

- Example for read AI1 current input
  Ex: Data 0x0030 = 48 in decimal. It means 0.48mA

Request		Response	
Field Name	(Hex)	Field Name	(Hex)
MODBUS Address	01	MODBUS Address	01
Function	03	Function	03
Starting Address Hi	00	Byte Count	02
Starting Address Lo	05	High Byte Data	00
Quantity of Outputs Hi	00	Low Byte Data	30
Quantity of Outputs Lo	01	High Byte CRC	B8
High Byte CRC	94	Low Byte CRC	50
Low Byte CRC	0B		

- Example for set Modbus Address and Baud Rate :

Request		Response	
Field Name	(Hex)	Field Name	(Hex)
MODBUS Address	01	MODBUS Address	01
Function	06	Function	06
Data Address Hi	00	Data Address Hi	00
Address Lo	00	Data Address Lo	00
High Byte Data	02	High Byte Data	02
Low Byte Data	06	Low Byte Data	06
High Byte CRC	08	High Byte CRC	08
Low Byte CRC	A8	Low Byte CRC	A8

Ex: High Byte Data 0x02 set Modbus Address as 02. The available rage is 1~247.

Low Byte Data 0x06 set Baud Rate as 38400

For 1 set baud rate as 2400

2 set baud rate as 4800

3 set baud rate as 9600

4 set baud rate as 14400

5 set baud rate as 19200

6 set baud rate as 38400

7 set baud rate as 115200

For MODBUS RTU CRC-16

```
/* The function returns the CRC as a type unsigned short. */
/* CRC Generation Function */

unsigned short CRC16 (puchMSG, usDataLen)

unsigned char *puchMsg：  /* message to calculate CRC upon */

unsigned short usDataLen：  /* quantity of bytes in message */
{
    unsigned char uchCRCHi = 0xFF; /* high CRC byte initialized */
    unsigned char uchCRCLo = 0xFF; /* low CRC byte initialized */
    unsigned uIndex; /* will index index into CRC lookup */
    while (usDataLen--) /* pass through message buffer */
    {
        uIndex=uchCRCHi ^ *puchMsgg++；  */ calculate the CRC */
        uchCRCHi = uchCRCLo ^auchCRCHi〔uIndex〕;
        uchCRCLo = auchCRCLo〔uIndex〕;
    }
    return（uchCRCHi<< 8 | uchCRCLo）;
}
```

High Order Byte Table

```
/* Table of CRC values for high - order byte */
static unsigned char auchCRCHi [ ] = {
0x00 , 0xC1 , 0x81 , 0x40 , 0x01 , 0xC0 , 0x80 , 0x41 , 0x01 , 0xC0 ,
0x80 , 0x41 , 0x00 , 0xC1 , 0x81 , 0x40 , 0x01 , 0xC0 , 0x80 , 0x41 ,
0x00 , 0xC1 , 0x81 , 0x40 , 0x00 , 0xC1 , 0x81 , 0x40 , 0x01 , 0xC0 ,
0x80 , 0x41 , 0x01 , 0xC0 , 0x80 , 0x41 , 0x00 , 0xC1 , 0x81 , 0x40 ,
0x00 , 0xC1 , 0x81 , 0x40 , 0x01 , 0xC0 , 0x80 , 0x41 , 0x00 , 0xC1 ,
0x81 , 0x40 , 0x01 , 0xC0 , 0x80 , 0x41 , 0x01 , 0xC0 , 0x80 , 0x41 ,
0x00 , 0xC1 , 0x81 , 0x40 , 0x01 , 0xC0 , 0x80 , 0x41 , 0x00 , 0xC1 ,
0x81 , 0x40 , 0x00 , 0xC1 , 0x81 , 0x40 , 0x01 , 0xC0 , 0x80 , 0x41 ,
0x00 , 0xC1 , 0x81 , 0x40 , 0x01 , 0xC0 , 0x80 , 0x41 , 0x01 , 0xC0 ,
0x80 , 0x41 , 0x00 , 0xC1 , 0x81 , 0x40 , 0x00 , 0xC1 , 0x81 , 0x40 ,
0x01 , 0xC0 , 0x80 , 0x41 , 0x01 , 0xC0 , 0x80 , 0x41 , 0x00 , 0xC1 ,
0x81 , 0x40 , 0x01 , 0xC0 , 0x80 , 0x41 , 0x00 , 0xC1 , 0x81 , 0x40 ,
0x00 , 0xC1 , 0x81 , 0x40 , 0x01 , 0xC0 , 0x80 , 0x41 , 0x01 , 0xC0 ,
0x80 , 0x41 , 0x00 , 0xC1 , 0x81 , 0x40 , 0x00 , 0xC1 , 0x81 , 0x40 ,
0x01 , 0xC0 , 0x80 , 0x41 , 0x00 , 0xC1 , 0x81 , 0x40 , 0x01 , 0xC0 ,
0x80 , 0x41 , 0x01 , 0xC0 , 0x80 , 0x41 , 0x00 , 0xC1 , 0x81 , 0x40 ,
0x00 , 0xC1 , 0x81 , 0x40 , 0x01 , 0xC0 , 0x80 , 0x41 , 0x01 , 0xC0 ,
0x80 , 0x41 , 0x00 , 0xC1 , 0x81 , 0x40 , 0x01 , 0xC0 , 0x80 , 0x41 ,
0x00 , 0xC1 , 0x81 , 0x40 , 0x00 , 0xC1 , 0x81 , 0x40 , 0x01 , 0xC0 ,
0x80 , 0x41 , 0x00 , 0xC1 , 0x81 , 0x40 , 0x01 , 0xC0 , 0x80 , 0x41 ,
0x01 , 0xC0 , 0x80 , 0x41 , 0x00 , 0xC1 , 0x81 , 0x40 , 0x01 , 0xC0 ,
0x80 , 0x41 , 0x00 , 0xC1 , 0x81 , 0x40 , 0x00 , 0xC1 , 0x81 , 0x40 ,
0x01 , 0xC0 , 0x80 , 0x41 , 0x01 , 0xC0 , 0x80 , 0x41 , 0x00 , 0xC1 ,
0x81 , 0x40 , 0x00 , 0xC1 , 0x81 , 0x40 , 0x01 , 0xC0 , 0x80 , 0x41 ,
0x00 , 0xC1 , 0x81 , 0x40 , 0x01 , 0xC0 , 0x80 , 0x41 , 0x01 , 0xC0 ,
0x80 , 0x41 , 0x00 , 0xC1 , 0x81 , 0x40
} ;
```

Low Order Byte Table

```
/* Table of CRC values for low - order byte */
static char auchCRCLo〔〕= {
0x00 ， 0xC0 ， 0xC1 ， 0x01 ， 0xC3 ， 0x03 ， 0x02 ， 0xC2 ， 0xC6 ， 0x06 ，
0x07 ， 0xC7 ， 0x05 ， 0xC5 ， 0xC4 ， 0x04 ， 0xCC ， 0x0C ， 0x0D ， 0xCD ，
0x0F ， 0xCF ， 0xCE ， 0x0E ， 0x0A ， 0xCA ， 0xCB ， 0x0B ， 0xC9 ， 0x09 ，
0x08 ， 0xC8 ， 0xD8 ， 0x18 ， 0x19 ， 0xD9 ， 0x1B ， 0xDB ， 0xDA ， 0x1A ，
0x1E ， 0xDE ， 0xDF ， 0x1F ， 0xDD ， 0x1D ， 0x1C ， 0xDC ， 0x14 ， 0xD4 ，
0xD5 ， 0x15 ， 0xD7 ， 0x17 ， 0x16 ， 0xD6 ， 0xD2 ， 0x12 ， 0x13 ， 0xD3 ，
0x11 ， 0xD1 ， 0xD0 ， 0x10 ， 0xF0 ， 0x30 ， 0x31 ， 0xF1 ， 0x33 ， 0xF3 ，
0xF2 ， 0x32 ， 0x36 ， 0xF6 ， 0xF7 ， 0x37 ， 0xF5 ， 0x35 ， 0x34 ， 0xF4 ，
0x3C ， 0xFC ， 0xFD ， 0x3D ， 0xFF ， 0x3F ， 0x3E ， 0xFE ， 0xFA ， 0x3A ，
0x3B ， 0xFB ， 0x39 ， 0xF9 ， 0xF8 ， 0x38 ， 0x28 ， 0xE8 ， 0xE9 ， 0x29 ，
0xEB ， 0x2B ， 0x2A ， 0xEA ， 0xEE ， 0x2E ， 0x2F ， 0xEF ， 0x2D ， 0xED ，
0xEC ， 0x2C ， 0xE4 ， 0x24 ， 0x25 ， 0xE5 ， 0x27 ， 0xE7 ， 0xE6 ， 0x26 ，
0x22 ， 0xE2 ， 0xE3 ， 0x23 ， 0xE1 ， 0x21 ， 0x20 ， 0xE0 ， 0xA0 ， 0x60 ，
0x61 ， 0xA1 ， 0x63 ， 0xA3 ， 0xA2 ， 0x62 ， 0x66 ， 0xA6 ， 0xA7 ， 0x67 ，
0xA5 ， 0x65 ， 0x64 ， 0xA4 ， 0x6C ， 0xAC ， 0xAD ， 0x6D ， 0xAF ， 0x6F ，
0x6E ， 0xAE ， 0xAA ， 0x6A ， 0x6B ， 0xAB ， 0x69 ， 0xA9 ， 0xA8 ， 0x68 ，
0x78 ， 0xB8 ， 0xB9 ， 0x79 ， 0xBB ， 0x7B ， 0x7A ， 0xBA ， 0xBE ， 0x7E ，
0x7F ， 0xBF ， 0x7D ， 0xBD ， 0xBC ， 0x7C ， 0xB4 ， 0x74 ， 0x75 ， 0xB5 ，
0x77 ， 0xB7 ， 0xB6 ， 0x76 ， 0x72 ， 0xB2 ， 0xB3 ， 0x73 ， 0xB1 ， 0x71 ，
0x70 ， 0xB0 ， 0x50 ， 0x90 ， 0x91 ， 0x51 ， 0x93 ， 0x53 ， 0x52 ， 0x92 ，
0x96 ， 0x56 ， 0x57 ， 0x97 ， 0x55 ， 0x95 ， 0x94 ， 0x54 ， 0x9C ， 0x5C ，
0x5D ， 0x9D ， 0x5F ， 0x9F ， 0x9E ， 0x5E ， 0x5A ， 0x9A ， 0x9B ， 0x5B ，
0x99 ， 0x59 ， 0x58 ， 0x98 ， 0x88 ， 0x48 ， 0x49 ， 0x89 ， 0x4B ， 0x8B ，
0x8A ， 0x4A ， 0x4E ， 0x8E ， 0x8F ， 0x4F ， 0x8D ， 0x4D ， 0x4C ， 0x8C ，
0x44 ， 0x84 ， 0x85 ， 0x45 ， 0x87 ， 0x47 ， 0a46 ， 0x86 ， 0x82 ， 0x42 ，
0x43 ， 0x83 ， 0x41 ， 0x81 ， 0x80 ， 0x40
} ;
```